High-Level Abstractions in Wireless Sensor Networks: Status, Taxonomy, Challenges, and Future Directions

Abrar M. Alajlan, Khaled M. Elleithy, *Member, IEEE*

Abstract— **Wireless sensor networks (WSNs) have gained a lot of considerations in recent years and have significant impacts on different application areas. Wireless sensors have been successfully deployed in different computing environments to measure, gather and process the raw information in the sensing area to the observers. Sensor networks provide infinite opportunities, but at the same time pose rough challenges due to the sensors' characteristics and the operating conditions of these sensors. This paper provides an extensive study of the current state-of-art in programming wireless sensor network, presenting a classification of programming levels in the field and highlighting some likely programming challenges and research future directions.**

***Index Terms*— Macroprogramming, Programming Approaches, Programming Challenges, Sensor Network.**

I. INTRODUCTION

Wireless sensor networks (WSNs) have gained a lot of considerations in recent years and have significant impacts on different application areas. They are composed of tiny embedded devices, each of which has radio transceiver to send or receive packets, processor to schedule and perform tasks, and power source to provide energy for the sensor [1]. Most often, WSN is utilized for the ease of deployment and enhanced flexibility of the network. Furthermore, it supports low cost dense monitoring of hostile environments as well as disaster relief, medical care and military surveillance [2].

The advantage of being able to place remote sensing nodes without having to run wires and the cost related to it is a huge gain. As the size of the circuitry of WSNs is becoming smaller along with the lower cost, the chances of their field of applications are significantly growing [3].

Manuscript received February 9, 2014.

A. Alajlan is with the Department of Computer Science and Engineering, University of Bridgeport, Bridgeport, CT, USA (phone: 2039082241; e-mail: aalajlan@bridgeport.edu)

K. Elleithy is with the Department of Computer and Electrical Engineering, Faculty of Engineering, University of Bridgeport, CT 06604, USA (e-mail: elleithy@bridgeport.edu).

Several programming approaches have been proposed to assist WSNs programming. Two broad classes of WSNs programming models have been explored lately; local behavior and global behavior abstraction [4]. In local behavior abstractions, the application has to be programmed in details at the node-level and the programmers need to synchronize the program flow between the sensing nodes and maintain the routing code manually. In contrast, global behavior abstractions or equivalently "High-level abstraction" has emerged as one of the most important aspects in sensor networks where it is applied to hide the internal operations from system programmers. The main objective behind high-level approach is the ability to treat a group of sensors or the entire network as one single unit rather than programming each node individually [5].

The main contribution of this work is to provide an extensive survey on taxonomy of programming approaches for wireless sensor networks. Our work also captures the programming requirements and uses them to evaluate each of the programming models. This paper also covers some open problems and challenges that need further investigation to make wireless sensor programming reaches its best level of performance and makes it highly usable and efficient.

Section II, identifies the requirements for sensor network programming. Section III, provides taxonomy on programming approaches for WSNs. An in-depth look on each level of the programming approaches is presented in Section IV, V and VI. Analysis and evaluation of each model is discussed in Section VII. Section VII investigates research challenges and future direction of programming WSNs. Conclusion in provided in Section IX.

II. REQUIREMENTS FOR SENSOR NETWORK PROGRAMMING

It is obvious that sensor networks can be used in multiple applications that can be deployed in diverse environments. Moreover, it is very easy to modify the internal functionality of sensor networks to perform different tasks to support many sensor network applications. In this section we discuss important requirements for sensor network programming.

A. Scalability

Many sensor network applications deploy hundreds or even thousands of nodes collaborating to achieve desired goal(s);

thus, scalability is one of the major designing attributes in sensor networks applications [6]. A scalable sensor network is representing the ability of the network to maintain its performance even when the network size has changed [7]. In WSNs scalability can be defined in two terms; size and geography. Scalability with respect to size states that if the application works properly with a few nodes, it can perform well with thousands of nodes. On the other hand, the scalability with respect to geography is defined as the ability to perform correctly in different geographical areas under different environmental conditions [8]. Since we cannot predetermine the location of sensor nodes and we cannot assure the lifetime of sensor nodes, the programming model should help programmers in such a way to design scalable applications that are able to deliver accurate results [7].

B. Localization

In wireless sensor network applications there are hundreds of nodes deployed in some areas such as underwater, in the middle of desert, or in inaccessible terrains, so their locations are random and unknown [9] , [10]. Thus, localization in sensor network, the determination of the geographical locations of sensors, is one of the important aspects for sensor network programming [11]. Many localization techniques have been proposed recently, either by deploying self-localized technique or by installing a Global Positioning System (GPS) device in each node to determine the exact location of the sensor node.

C. Failure-Resilience

Failure –resilience or (Fault-tolerance) is one of the most challenging requirements in programming wireless sensor networks [12]. Sensors are usually deployed in inaccessible terrains where people cannot reach the sensor nodes at that place. Some nodes might fail due to the resources limitation, hardware fault or it could be an intrusion from attackers. The failed sensors may lead to inefficient functioning of the network [13].

Thus, the system should keep performing properly even after unreliable communication, node failures, link failures, or unavailability of the network due to misbehaving nodes [14][15].

D. Energy-Efficiency

Energy efficiency is one of the most important issues in designing sensor networks. The overall design of sensor networks should mainly emphasize on enhancing the performance in terms of reduced power consumption.

The total lifetime of a battery-powered sensor networks is limited by the non-rechargeable battery's capacity and each sensor node is equipped with a limited computation processor to perform its task [16].

Energy efficiency is very important factor in developing WSNs applications especially for continuous monitoring applications such as disaster monitoring, military surveillance and remote patient monitoring, etc. [17].

Thus, the programming model for sensor networks should deploy some applications that attain a proper level of energy-efficiency and able to deliver demanded results [18].

E. Collaboration

Collaboration is another important characteristic of wireless sensor applications. WSNs applications have been growing recently. These applications vary in terms of size and number of nodes, from large scale networks to the small ones. All nodes in one application need to communicate in such a way so that the data from these sensors are gathered and analyzed. Thus, collaboration between sensor nodes is essential for these sensors to cooperatively and effectively work together to complete the desired tasks [19] [20].

Collaboration is not an independent requirement, it can support other requirements. For instance, collaboration between sensor nodes may reduce the failure-resilience where the sensing process remains functional even after one node failed.

F. Time Synchronization

Time-synchronization between nodes is another essential requirement for sensor programming execution. Many WSNs applications such as tracking application and implementation of TDM require a timer synchronization that is maintained at each sensor node [21].

Clock synchronization is a process used to ensure an accurate scheduling between nodes with no collision [22]. Moreover, WSNs have limited power therefore; time-synchronization technique helps to reduce the power consumption by passing some nodes off from time to time [23].

III. PROGRAMMING APPROACHES FOR WSNS: A TAXONOMY

In this section we present taxonomy of the programming high- level approaches for WSNs. Figure 1 depicts the entire taxonomy that categorize the wireless sensor network programming high-level approaches into group level and network level abstractions.

High-level programming approach mainly focuses on simplifying the collaboration between sensor nodes.

One approach is to divide the whole network into a set of groups and treat each group as a single entity which is called “Group-level abstractions”. It helps the programmer to describe collaborative algorithms easily. This approach is further divided into physical groups and logical groups. In physical group, the network can be grouped based on the physical location of the node, whereas the logical group is based on the shared properties among nodes.

The other approach of high-level abstraction is network level abstraction or “macroprogramming abstractions” where the whole network is treated as a single entity. It is an application centric-view, thus, it helps the programmer to focus on the programming logics rather than programming the platforms.

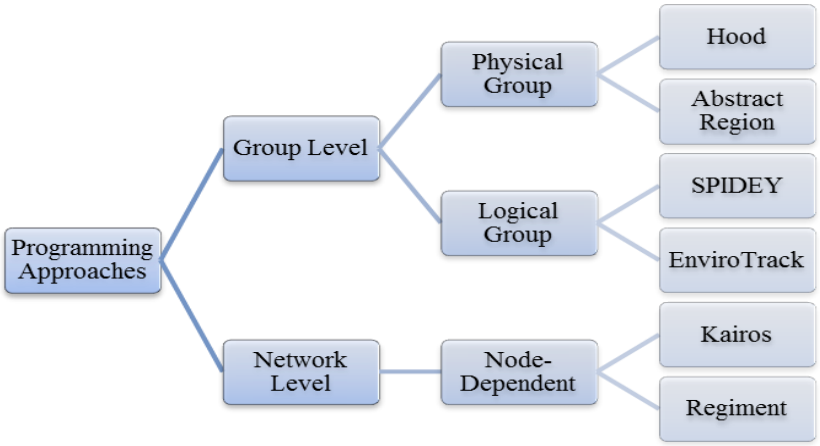

Fig 1: A Taxonomy of programming global behavior approaches in wireless sensor networks

IV. Group-Level Abstraction

The main concept behind a group-level abstraction in WSNs is to divide the whole network into small groups and perform computations on those groups instead of dealing with each single node. In a group- level abstraction, the network can be grouped based on the physical locations of the nodes (Neighborhood Based) or it can be grouped logically [24].

A. Physical Group

The notion of physical group or "neighborhood based group" is basically a node with its neighbor's without paying any attention to the properties of these nodes [18]. This technique is used to hide the communication details between the nodes and it can be used in "localized algorithms" where the interaction between participating nodes is limited to their neighbors as in [25].

Hood

Hood is one example of a neighborhood- based programming abstraction where a given node is limited to communicate and share data with neighboring nodes only. This physical closeness is determined by the physical distance or the number of hops between the sensor nodes [26].

In Hood, all nodes in each group have to be in the same network and if one node moves to another network then it is not a member of that group.

Abstract Region

Another example of a neighborhood-based group abstraction is Abstract Region which relies on the concept of grouping the nodes in mesh, spanning tree or could be based on the geographic locations of these nodes as shown on Figure 6 [27]. Abstract Regions as in Hood, cannot group nodes from different network. Moreover, this model can be adapted within different network conditions to attain different levels of energy and bandwidth usage as well as the accuracy level of shared operations. Also, each region is separated from other regions and requires a specific implementation.

A. Logical group

A logical group abstraction can be defined as a set of nodes that share the same properties in sensor networks such as node types, sensor inputs, or perform the same tasks [14]. Unlike neighborhood based, the logical group is considered to be a dynamic group since it is based on the shared properties and not limited by the physical location of nodes [28].

Logical group-based, cannot cross multiple networks at the same time which means we cannot reuse the existing sensors without reprogramming them [29].

EnviroTrack

One example of Logical based group is EnviroTrack. It is an application used to track programs where a set of nodes that detect the same event are grouped together [30].

SPIDEY

Another example of logical based group is a SPIDEY language where a set of nodes are grouped based on their shared properties [26]. In SPIDEY language, each node has both static and dynamic attributes which are used to determine the nodes logical neighbors as in [26]. SPIDEY delivers communication APIs, where a broadcasted message is sent to a logical neighborhood instead of nodes that fall in the same communication range. This technique helps programmers to clearly specify the communication range and which nodes to select as a neighbor.

V. Network-Level Abstraction

Recently, several macro-programming abstractions have been introduced. Macro-programming systems or equivalently "networking abstractions" considered to be high-level WSN programming model where the whole sensors network is treated as a single system [14].

This approach helps the programmers to emphasize on improving the semantics of the program instead of studying the characteristics of the programming environments [14].

One type of macroprogramming is node- dependent abstraction where the programmers define the global behavior

of the entire system as a set of nodes that can be treated simultaneously in one program [4].

A. Node Dependent Approach

Node-dependent approach is intended to deliver more flexibility than group level approaches. This approach allows programmers to define the global behavior of the computation in terms of nodes and their states [31]

Kairos

Kairos is a node-dependent abstraction where the neighboring nodes can be computed in parallel and communicate using common requests at specific nodes [32]. Kairos has a centralized programming environment which is translated later by the compiler to many executable effective nodal programs [32]. Kairos enhances the use of sensor programming languages by providing three simple mechanisms: node abstraction, and accessing data on a remote node [31].

Kairos implements an eventual consistency method; by adopting this feature the program is able to deliver the most accurate result even if an internal node is not assured to be reliable. Thus, Kairos can be used with many well-known programming languages such as python as in [32].

Regiment

Another example of node-dependent abstraction is Regiment, a purely macroprogramming functional language that allows the direct use of program state [2].

However, it uses what is called monads; described in more detail elsewhere in [33]. In Regiment, programmers deploy groups of data stream or what is called signals. These signals used to represent the finding of each individual node.

Moreover, Regiment applies a multi-stage programming mechanism to support the use of different programming languages that are not maintained by the given program [34]. Also, Regiment enables the use of generics that qualify the program to pass any data types as in C++.

VI. Analysis and Evaluation

In this section, we focus on the most important strategies that are used in each programming model to fulfil the programming requirements discussed earlier.

A summary of how each level of the programming approaches addresses these requirements is shown in the next three tables below.

Table I shows the evaluation of macroprogramming approach based on the programming requirements. The main approach to satisfy scalability is to reduce the communication between the sensor nodes. Regarding to localization, Regiment provides the ability to divide the tested area to spatial regions to facilitate the localization and communication processes. Also, Regiment is resilient to failure where the master node or (Anchor) in each region is responsible to cover if a node fails or loses connectivity to others [2]. In addition, Kairos adopts an eventual consistency method to deliver the most accurate result even if an internal node is not assured to be reliable. Kairos also uses a cashing technique to reduce the communications and power consumption.

Table I: Evaluation of macroprogramming models for sensor networks

Evaluation Factor	**Network-Level**	
	Node-Dependent	
	Kairos	***Regiment***
Scalability	No evidence for support	Purely functional language. Permit the use of fold, map functions.
Localization	Each node is only responsible for localizing itself	Use Region for the purpose of localizing sensing
Failure-Resilience	Eventual consistency	Anchor " leader" is an object persists across node failures
Energy-Efficiency	Caching	Purely functional language. Permit the use of fold, map functions
Collaboration	Implicitly express both distributed data flow and control flow.	Capable of expressing groups of nodes with geographical, and logical relationships
Time Synchronization	Automatically synchronizes nodes when a checkpoint is taken or restored.	Use signals to represent the finding of each individual node.

Table II shows how the programming requirements implemented in each programming model at the group level abstraction. Since all the programming models listed on table II are group based abstractions, the scalability, collaboration and data aggregation are supported through data sharing [14]. Caching technique is used in several programming models at this level to reduce the communications between sensing nodes and helps to save energy [27] [30].

Caching and abstract region are employed in Hood to improve the communication failures by replacing the failed data with the old cached one. [14].

However, SPIDEY utilizes redundancy mechanism to avoid flooding the whole program and to limit the propagating of information [32].

There are some components or functions attached to each programming model to improve localization: Hood uses mirror to reflect node locations or time synchronization services [30]. In this case, abstract region starts with neighbor discovery where each node initiates the process of discovering the location of its neighbors [27]. As the tracked objects move in EnviroTrack, the location of participating nodes has to be known by using some functions like *Location*: avg (position) [30]

From the previous analysis we can conclude the all these requirements are dependent and related to each other. Indeed, each single requirement can support or work in favor of other requirements. For example, data sharing through multiple nodes improves the scalability, collaboration and also energy-efficiency.

Moreover, energy-efficiency and scalability is also can be improved by reducing the communication between sensing nodes.

VII. FUTURE RESEARCH DIRECTIONS AND PROGRAMMING CHALLENGES FOR WSNS

As stated earlier, many programming approaches have sought the development of programming support to address the programming requirements of WSNs. In this section, we highlight some open problems and challenges that need further investigation to make wireless sensor programming reaches its best level of performance and makes it highly usable and efficient.

A. Reprogramming

Several programming approaches have been introduced and discussed in the past decades. However, there are many programming challenges still unresolved and need further study to make the WSNs programming valuable and effective ; thus, in this section we list some of them and discus the future direction of programming WSNs Reprogramming

The network programming requirements might change over time, and this change could be parameter changes or reprogram the entire system. Also, wireless sensors might move from one network to another, but the limited resources of these sensors may result in short-lived systems.

Thus, sensing nodes should have a dynamic reconfiguration services to keep these sensors functional for a long time [35].

Table II: Evaluation of group level abstractions in WSNs

Evaluation Factor	**Group-Level**			
	Physical Group		***Logical Group***	
	Hood	***Abstract Region***	***EnviroTrack***	***SPIDEY***
Scalability	Supported through data sharing	Supported through data sharing etc.	Supported through data sharing etc.	Supported through data sharing etc.
Localization	Use mirrors to reflect location.	Done at neighbor discovery stage	As the tracked objects move, the location of have to be known.	physical location of each node should be identified when creating node.
Failure-Resilience	Caching :substitute the data with old cached data	Caching: substitute the data with old cached data	Dynamic group management and leader election	Utilize redundancy mechanism
Energy-Efficiency	Power consumption supported through data sharing.	Supported through data sharing Caching low-level control knobs.	through data sharing Caching ("freshness threshold")	Supports aggregation at group level through data sharing etc.
Collaboration	Asymmetric Group definition and operations on neighbor	Group definition and operations on group. Operation include enumeration using iterator and MPI-like reduction	Dynamic group definition and operations on group	Group definition and operations on group
Time Synchronization	Use mirror for time synchronization	Use timeout mechanism,	Timer handler to executes one iteration at time	Contains a time-period attribute at each node.

In WSNs, the basic form of heterogeneity is deploying multiple different types of sensors in one application, each of which performs different task and has different energy and resources.

Heterogeneity in a WSN is used to improve the overall reliability and lifetime of the network [36].

Heterogeneity in WSNs has two forms: physical heterogeneity and logical heterogeneity.

From programming point view, how to deploy heterogeneous sensors efficiently and how to program the entire system with these sensors are the main concerns in developing WSNs applications.

C. Quality of Service

Quality of service is one of the important challenges in designing wireless sensors applications. As stated earlier, wireless sensors are equipped with limited energy resources. Accordingly, system designers need to balance between energy consumed and some quality services such as accuracy and error rates to get efficient results with a satisfying quality. Quality is a very crucial element in designing sensor network application since there are certain actions will be taken according to the sensed result. [14].

The above requirements and the demanding deployment environment of wireless sensors make sensor programming the most challenging task in developing wireless sensors applications. In spite of the considerable effort carried out to let WSN programming model reach its best level of performance, still there are several open problems that need further investigation to make wireless sensor programming highly usable and efficient.

VIII. CONCLUSION

In this paper, we have provided taxonomy of programming high-level abstractions in wireless sensor networks. Two different levels of programming approaches have been discussed: group level and network level. Several examples have been covered and evaluated based on some programming requirements for each level. Designing efficient programming models for WSNs has many challenges to overcome such as reprogramming, heterogeneity, and quality of service. Research must carry on in all capacities of sensor networks programming model to address these challenges and we believe that advances in WSNs programming models will facilitate deploying energy efficient, reliable, and accurate applications in the WSNs domain.

REFERENCES

[1] M. Tubaishat and S. Madria, "Sensor Networks: An Overview," *IEEE Potentials*, vol. 22, no. 2, pp. 20–23, April/May 2003.

[2] R. Newton and M.Welsh, "Region Streams: Functional Macroprogramming for Sensor Networks," *Proceeedings of the 1st international workshop on Data management for sensor networks: in conjunction with VLDB*, 2004, pp. 78-87

[3] N.Marriwala and P.Rathee, "An Approach To Increase The Wireless Sensor Network Lifetime," *IEEE on Information and Communication Technologies*, Oct.2012, pp. 495-499

[4] S.Choochaisri, N. Pornprasitsakul, C.Intanagonwiwat, "Logic Macroprogramming for Wireless Sensor Networks," *International Journal of Distributed Sensor Networks*, 2012

[5] U.Bischoff and Kortuem, G., "A State-based Programming Model and System for Wireless Sensor Networks", *IEEE International Conference on Pervasive Computing and Communications*, March, 2007, pp. 19-23

[6] K. Akkaya and M. Younis, "A Survey on Routing Protocols for Wireless Sensor Networks*," **Ad Hoc Networks***, vol. 3, no.3, May.2005, pp. 325–349

[7] M. Sousa, A. Kumar, M. Alencar , W. Lopes, "Scalability in An Adaptive Cooperative System for Wireless Sensor Networks," *IEEE International Conference on Ultra-Modern Telecommunications Workshops (ICUMT)* , Oct. 2009

[8] R. Venkateswarlu and D. Janakiram, "A Simple Model for Evaluating The Scalability in Wireless Sensor Networks," *IEEE International Conference on Intelligent Sensors, Sensor Networks and Information Processing Conference*, Dec. 2005

[9] N. Patwari, J. N. Ash, S. Kyperountas, A. O. Hero, R. L. Moses, and N. S. Correal, "Locating The Nodes: Cooperative Localization in Wireless Sensor Network," *IEEE Signal Processing Magazine*, vol. 22, July. 2005, pp. 54–69

[10] Z. Chaczko, R. Klempous, J. Nikodem, M. Nikodem, J. Rozenblit, An Improvement of Energy Aware Routing in Wireless Sensors Network, *European Modeling and Simulation Symposium*, Barcelona, Oct. 2006

[11] T. Yasuhisa,"Node Localization for Sensor Networks using Self-Organizing Maps", *IEEE Topical Conference on Wireless Sensors and Sensor Networks (WiSNet)*, Jan 2011, pp. 61-64

[12] M.Bellalouna A. Ghabri, "A Priori methods for Fault Tolerance in Wireless Sensor Networks," *IEEE World Congress on Computer and Information Technology (WCCIT)*, June. 2013

[13] L.M. Wang, J.F. Ma, C. Wang, and A.C. Kot, "Fault and Intrusion Tolerance of Wireless Sensor Networks," *IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, April. 2006

[14] R. Sugihara and R.Gupta, "Programming Models for Sensor Networks: A Survey," *ACM Transactions on Sensor Networks*, vol. 4, no. 2, Article 8, March 2008

[15] D.Geetha, N. Nalini, R. Biradar, "Active Node based Fault Tolerance in Wireless Sensor Network," *Annual IEEE India Conference (INDICON)*, Dec. 2012, pp. 404 – 409

[16] A. Alajlan, B. Dasari, Z. Nossire, K. Elleithy and V. Pande, "Topology Management in Wireless Sensor Networks: Multi-State Algorithms," *International Journal of Wireless & Mobile Networks (IJWMN)*, vol 4, no. 6, Dec 2012 pp. 17-26

[17] D.Chaudhary, and L. Waghmare, "Energy Efficiency and Latency Improving Protocol for Wireless Sensor Networks," *IEEE International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, Aug. 2013 pp. 1303 – 1308

[18] K. Vardhe ,C. Zhou, D. Reynolds, "Energy Efficiency Analysis of Multistage Cooperation in Sensor Network," *IEEE GLOBECOM*, Dec. 2010, pp. 1 - 5

[19] M. Huang, and Y. Hen Hu "Collaborative Sampling in Wireless Sensor Networks."*IEEE GLOBECOM* Dec.2010, pp. 1-5

[20] W. Li, J. Bao, and W. Shen," Collaborative Wireless Sensor Networks: A Survey," *IEEE SMC* Oct. 2011, pp. 2614 –2619

[21] S. Lee, U. Jang and J. Park, "Fast Fault-Tolerant Time Synchronization for Wireless Sensor Networks," *IEEE ISORC* May.2008, pp. 178 – 185

[22] K. Yaguang, Z. Xifang, C. Huakui, " Intelligent Time Synchronization in Sensor Network", *IEEE International Conference on Wireless, Mobile and Multimedia Networks*, Nov. 2006, pp. 1 - 4

[23] S. Lasassmeh and J.Conrad, "Time Synchronization in Wireless Sensor Networks: A Survey," IEEE SoutheastCon, Mar. 2010, pp. 242 – 245

[24] H.Paul and T. *McGregor, "*Wireless sensor networks: a distributed operating systems approac*h," Proceedings of New Zealand Computer Science Research Student Conference, NZCSRSC*, 2009

[25] D. Estrin, R.Govindan, J.Heidemann, "Next century challenges: scalable coordination in sensor networks," *Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking*, 1999. pp. 263 270

[26] L. Mottola and G. P. Picco "Logical Neighborhoods: A Programming Abstraction For Wireless Sensor Networks", *Proc. 2nd International Conference on Distributed Computing on Sensor Systems (DCOSS)*, 2006. pp.150 -168

[27] M.Welsh and G.Mainland, "Programming Sensor Networks Using Abstract Regions," *Proceedings of the 1st conference on Symposium on Networked Systems Design and Implementation*, vol.1, 2004

[28] L.Mottola and G. P. Picco ," Programming Wireless Sensor Networks with Logical Neighborhoods, " *In Proceedings of the 1st ACM International Conference on Integrated Internet Ad-hoc and Sensor Networks (INTERSENSE)*, May 2006.

[29] P.Vicaire, E.Hoque, Z.Xie," Bundle: A Group-Based Programming Abstraction for Cyber-Physical Systems," *IEEE Transactions on Industrial Informatics*, vol.8, no.2, May 2012, pp. 379 – 392

[30] T. Abdelzaher. et. al. "EnviroTrack: towards an environmental computing paradigm for distributed sensor networks," *Proceedings of the 24th International Conference on Distributed Computing Systems*. 2004, pp. 582–589.

[31] R. Gummadi, O. Gnawali, and R. Govindan, "Macro-programming Wireless Sensor Networks using Kairos," *Proceedings of the International Conference on Distributed Computing in Sensor Systems (DCOSS*) June 2006

[32] L. Mottola and G. P. Picco, "Programming Wireless Sensor Networks: Fundamental Concepts and State Of The Art," *ACM Computing Surveys*, vol. 43, no. 3, 2011

[33] E. Moggi,"Computational Lambda-Calculus And Monads," *Proceedings. Fourth Annual Symposium on Logic in Computer Science*, June.1989, pp. 14 – 23

[34] R. Newton, G. Morrisett, M. Welsh, "The Regiment Macroprogramming System," *International Symposium on Information Processing in Sensor Networks*, April.2007, pp. 489 – 498

[35] M. Rossi, G. Zanca, L. Stabellini, R. Crepaldi, A. F. H. III, and M. Zorzi, "SYNAPSE: A Network Reprogramming Protocol for Wireless Sensor Networks using Fountain Codes," *in Proc. of IEEE SECON*, June.2008, pp. 188 – 196

[36] B. Rubio, M. Diaz and J. Troya, "Programming Approaches and Challenges for Wireless Sensor Networks*," IEEE Second International Conference on Systems and Networks communications (ICSNC)*, Aug. 2007

Abrar M. Alajlan: is a Ph.D. candidate in computer science and engineering at the University of Bridgeport with research interests primarily in wireless sensor networks. Her work focuses on programming the wireless sensor networks and deploys them in hostile environments. Her most recent working paper explores a new programming model to simulate wireless sensor networks to find the best routing path. Abrar Alajlan graduated from Troy University with a master's degree in MBA with a concentration in Information Systems (IS) from Troy University, Troy, AL in 2011. She received a BS in Computer Science from Umm Al-Qura University, Makkah, Saudi Arabia

Dr. Khaled Elleithy is the Associate Dean for Graduate Studies in the School of Engineering at the University of Bridgeport. He has research interests are in the areas of network security, mobile communications, and formal approaches for design and verification. He has published more than two hundreds research papers in international journals and conferences in his areas of expertise.

Dr. Elleithy is the co-chair of the International Joint Conferences on Computer, Information, and Systems Sciences, and Engineering (CISSE). CISSE is the first Engineering/Computing and Systems Research E-Conference in the world to be completely conducted online in real-time via the internet and was successfully running for six years. Dr. Elleithy is the editor or co-editor of 12 books published by Springer for advances on Innovations and Advanced Techniques in Systems, Computing Sciences and Software.